From: ALAN B. CONLEY [aconley@cisco.com]
Sent:

MANAGING AD REPLICATION TOPOLOGY

Introduction

In order for Active Directory to operate effectively, the replication process needs to be managed. AD uses the concept of sites and site links for describing the replication topology. Sites are collections of "well connected" subnets. Well connected generally means high speed LAN connections.
Site links are connections between sites, with an associated cost. Sites are used in the user logon process and for AD replication.

* Sites are a group of subnets. Multiple subnets can be represented by a single high level network prefix (aka "address block").
* Sites allow clients to find the closest DC, GC, DFS share point or application distribution point (via SMS).
* Whenever a user logs onto the network, NT will attempt to locate a domain controller in the same site as the user.
* Sites are used to plan AD replication. They can be used to control the rate of replication, as well as the frequency.
* Default Placement. When ADS is installed for the first time in a Win2K environment, a default site will be created. It will be named "Default-First-Site-Name". This site can be renamed.

The topology created by generating sites and site links is known as the AD replication topology and is stored in the directory. A process needs to be developed in order to create and maintain the AD replication topology.

Problem Statement

Although AD includes a tool for managing sites and site links, the process is largely manual. Considering the size of the Cisco
network and the frequency of IP address renumbering, this process would be both labor intensive and error prone. Therefore, automating the creation and maintainence of AD topology is highly desireable.

Proposed Solution

Sites are very much like LANs. Site links are very much like WAN links between LANs. Additionally, site links have associated costs, which are similar to routing metrics. Therefore, it
should be possible to automatically generate the AD site and
site link data, based on network topology. There are a couple of ways this could be done, but probably the most expediant would be to parse router configuration files for the required information. The process would need a mechanism to override parts of the automatically generated topology if desired and also a method for injecting the topology into AD.

Requirements

Automating the generation of site and site link data based on router configuration files will have the following requirements.

* Process will need to run at least once a day.
* Will need the ability to override or ignore topology information derived from the router configuration files.
* Need to devise a methodology for determining which subnets constitute a site.
* Need to define what constitues a link.
* Need to be able to automatically inject topology information into AD.
* Accurate data will depend on having current configuration files for all IT supported routers.
* Program will need to be able to generate costs for each site link.
* Subnets within a site should be aggregated under a larger prefix (aka address block).
* NT Ops will need to generate a "standard" list of AD site names. Site names will be maintained in EMAN.
* Need a mechanism to link sites generated by the program to standard site names maintained by IT networking.
* There will need to be some mechanism available to handle "managed router" services, where the router configuration files are unavailable.
* Exception handling. The automated process will generate various exceptions. A process will need to be implemented that will ensure exceptions are reviewed and resolved.
* A consistant and standardized method for naming sites and links is required.

Responsibilities

* NT Ops is responsible for maintaining the list of AD site names.
* NT Ops is responsible for adding DCs to EMAN as NT hosts and associating them with the appropriate site names.
* GCTS Networking and NT Ops are responsible for resolving exceptions generated by the replication topology generation program.
* EMAN is responsible for developing and maintaining the replication topology generation program.
* GCTS Networking is responsible for maintaining preprocessing data that will be used to override topology resulting from processing the router configuration files.

DESIGN DETAILS

30,000 ft View

* A list of AD site names are maintained in EMAN by NT Ops.
* NT Ops ensures DCs are resolvable via DNS.
* NT servers are added to EMAN by NT Ops. One of the specified pieces of information is the appropriate AD site name.
* repl_gen runs once every 24hrs.
  - Reads in preprocessing information that may be used to override topology generated by reading router configuration files.
  - Reads in the most recent version of router configuration files.
  - Based on various commands, sites and site links are generated.
  - Bandwidth on WAN links is used to generate appropriate AD replication costs.
  - The result of this process is a replication topology suitable for both the configuration and schema NCs.
  - A variety of exception conditions are checked. Exception reports are generated and will need to be reviewed.
  - Under some conditions, the process may abort in which case the replication topology will not be updated.
* Should repl_gen complete successfully, the resultant toplogy will be inserted into a specified DC.
* In order to prevent partial topology information from being

replicated, the replication process on the DC will be suspended during the topology update.

Processing Router Config Files

In order to automatically generate site and site link information from router configuration files a methodology needs to be adopted to determine what constitutes sites and links. The method adopted will use router "interfaces" for this purpose.

* Sites are determined by subnets on LAN interfaces. LAN interfaces include:

  - FastEthernet
  - Ethernet
  - Fddi
  - Vlan
  - GigabitEthernet
  - TokenRing

* Site links are determined by WAN interfaces. WAN interfaces include:

  - ATM
  - POS
  - Serial
  - Hssi
  - FR-ATM
  - Tunnel
  - CBR

* Interfaces that are ignored include:
  - Loopback
  - Dialer
  - Null
  - Group-Async
  - Async
  - BRI
  - BVI
  - Multilink
  - Switch
  - Virtual-Template

* Ignore "Administratively Down" (aka shutdown) interfaces.
* Ignore any interfaces where the "ip address" line is of the format: ip address negotiated
* Need to handle secondary addresses.
* Can ignore all of these
  brussels-gb2.cisco.com:ip route 172.17.165.0 255.255.255.0 Null0 199
* No need to worry about ip unnumbered.

Access Networks (Home ISDN/Frame/DSL and Dial-in)

Most access networks do not appear in router configuration files. Typically, the configuration file for home routers is not available. Service routers usually have route summaries for home networks.

* Since home networks will not have AD servers, a home network will be part of the nearest site.
* How do you tell home networks from remote sales offices?
* A template with network prefixes that supported all home networks could be added.

1918 Addressing

Use of RFC 1918 address space in the corporate network should not be a problem. 1918 addresses that have been designated by networking as routable (such as those used for IP phones) will be treated no differently than any other portion of the internal routable address space. Those 1918 blocks which have been designated as non-routable lab space will be ignored by the process using a pre-processing block.

Pre Processing

---

* May want to tell site-gen to read from the database
  a list of "ignores" or preprocessed information. We provide an
  EMAN front end to site management that allows someone to associate
  one or more blocks with an AD site from the lookup table or
  with a "None" site if we want the block ignored.
  site_gen reads this information first. Any time it comes across
  this information in a router configuration file, it ignores
  the entries. You'd probably want to flag these sites somehow,
  so that you knew they were "manually" created.
* May want to be able to create site-links for some of the
  above generated sites.

```
# SITES
#   Format: Block|AD Site
#   10.0.0.0/23|RTP
#   10.0.0.x/23|SJ
#   10.0.0.y/23|IGNORE
# SITE LINKS
#   Format: SiteLink|Metric
#   SJ-RTP|5000
#
```

Definitions

---

* DC: Domain Controller (aka Active Directory Server). List
  of DCs are read in from EMAN. When a DC is entered into EMAN,
  the user associates it with a site-name-eman.

* site: A collection of subnets in the form of addresses and
  prefix lengths. i.e. 171.68.0.0/16

* site-gen: The name of the program that automatically generates
  sites and site-links.

* site-name-eman: The name taken from the EMAN ad_site lookup table.
* site-name-ip-route: The name given to a site that is generated
  from a set of "ip route" configuration file statements collected
  from a single router.
* site-name-remote: The site-name at the other end of a site-link
  from a site. Can be either a site-name or a site-name-temp.
* site-name-temp: The temporary name given to a site when it
  is created by the process described in this document.
  A site retains this name until it can be associated with
  a site_name.
* site-name-pre-proc: The name associated with a site which is
  read in during the pre processing stage.

* site-link: The connection between two sites. Typically represented
  by some type of serial link. A site-link has a name and a cost
  (site-link-cost).
* site-link-half: A site-link connected to a site, but for which
  the remote site has not been identified.
* site-link-pre-proc: A site-link connecting two site-names with
  an associated site-link-cost generated during the pre-processing

stage.
* site-link-cost: The cost/metric associated with a site-link based on bandwidth.
* site-block: An IP address block that results from summarizing as many subnet-lans or subnet-ip-routes in a site as possible.

* subnet-lan: An IP subnet generated from a LAN interface and associated with a site.
* subnet-link: An IP subnet generated from a LINK interface and associated with a site-link or a site-link-half.
* subnet-ip-route: An IP subnet generated from an "ip route " statement in a router configuration file.

* router-name: The name of a router taken from the router config file name.

* pre-proc-block: An address block generated during the preprocessing stage. These blocks are either associated with a site-name-pre-proc, or are specified to be ignored.

* island: a collection of sites that are connected by site-link

* abort: Print and exception, send a page exit site-gen.

Algorithm Outline

The following is a high level overview of the proposed algorithm for generating sites, links and link costs. Note that anytime a block comparison has to be done, the program will also have to verify that the block is a valid block (i.e. can't have 171.68.1.0/23).

* Stage 1
  Read in all preprocessing information.
  - Read in the list of site-name-emans from the EMAN ad_sites lookup table.
  - Read in the list of site-link-pre-procs and associated site-link-costs. (i.e. SJ-RTP|5000)
    + Parse the site names from the site-links and compare with the EMAN site-name-emans. If there is not a match, print an exception and discard the site-link.
    + Parse out the site-link-cost and verify that it is within the valid range. If not, print an exception.
  - Read in the list of pre-proc-blocks and their associated site-name-pre-procs (i.e. 171.68.10.0/24|AMS) or if they are to be ignored (i.e. 171.68.10.0/24|IGNORE)
    + If the site-name-pre-proc does not match a site-name-eman, print an exception and discard the pre-proc-block.
  - Read in the list of DCs from EMAN and their associated site-name-emans.
  - Use DNS to associate a DC name to an IP address to a site-name-eman. If a server fails resolution, generate an exception, discard the DC and continue processing. If more than 2 servers fail resolution, abort.
  - Use DNS to determine which domain each server is associated with. (i.e. dc.na.cisco.com => NA) If its not associated with any domain, print an exception and discard. (Need to verify that all DCs will be DNS servers. If not, we'll need to use a different method for determining domains. i.e. use LDAP query, or maybe store it in the DB.)
  - WHAT TO DO ABOUT CISCO.COM.
  - Compare the list of DC IP addresses with the pre-proc-blocks.
    + If a DC IP address falls inside a pre-proc-block, make sure the site-name-pre-proc associated with the pre-proc-block and the DC are the same. If not,

print an exception and abort.
  + If a DC IP address falls inside a pre-proc-block that is designated as "ignored", print an exception, but continue processing. Assume the "ignore" overrides the DC IP address and discard the DC from further processing.
- At this point, we have:
  + A list of site-name-emans.
  + A list of DC Names that map to IP addresses that map to site-name-emans and domain names.
  + A list of pre-proc-blocks that map to site-name-pre-procs.
  + A list of pre-proc-blocks that are to be ignored.
  + A list of site-link-pre-procs and associated site-link-costs.

* Stage 2
Read the router configuration files and a list of routers from EMAN.
- Select out all routers with a Support_Group of "ECS or IS" from EMAN. Also get the "download status".
- Compare the EMAN routers with the list of router configs.
  + If a router exists in EMAN, but there is no config file, generate an exception, even if it is configured as "No Download". Exception report will flag "No Downloads".
- Generate a router-name from the name of the config file.
- Check EMAN database for date of "last successful download".
  + If the date is more than three days old, print a warning, but keep the config.
  + If the date is more than seven days old, print an exception and discard the router's configuration.
- Create a site per router-name.
- Put all subnet-lans found on the router in the site.
- Create a site-name-temp for each site.
- Associate the router-name with the site-name-temp.
- Create a site-link-half for each LINK interface.
- Each site-link-half needs to be associated with the site-name-temp and a subnet-link.
- Create subnet-ip-routes based on "ip route" statements. Exclude Null0 and 0.0.0.0 routes.
- Associate each subnet-ip-route with the site-name-temp of the router. (Note this is just an association. These subnets are NOT to be merged with the site-name-temp subnet-lans).
- At this point, we have:
  + A list of site-name-emans.
  + A list of DC Names that map to IP addresses that map to site-name-emans and domain names.
  + A list of pre-proc-blocks that map to site-name-pre-procs.
  + A list of pre-proc-blocks that are to be ignored.
  + A list of site-link-pre-procs and associated site-link-costs.
  + A list of site-name-temps associated with:
    - A router-name.
    - A list of subnet-lans.
    - A list of site-link-halfs.
      + Each site-link-half is associated with a subnet-link.
    - A list of subnet-ip-routes.

* Stage 3
Clean up the "ip routes". Compare all subnet-ip-routes with all other subnet-ip-routes.'
- In order to reduce processing time, ignore all /29 - /32 blocks in the following tests. These should be fine anyway due to their small size. (May have to revisit later for at least block validity checks.)
- If a subnet-ip-route is a superset of another subnet-ip-route, mark the larger subnet-ip-route as to be discarded. To prevent printing the same exception many times, we'll just keep count of the number of times a subnet-ip-route meets this criteria and then print a single exception with a count.

- If two subnet-ip-routes are identical, mark the subnet as to be discarded due to identical subnet-ip-route. Keep this as a count as well.
- If two subnet-ip-routes overlap, mark the subnet as to be discarded due to overlap. Keep this as a count as well.
- When finished processing, print out two exception reports. One for supersets and one for identicals. Then discard all subnet-ip-routes that met this criteria.
- At this point, we have:
  + The same results as at the end of Stage 2, with the exception that some of the subnet-ip-routes have been discarded.
  + A list of site-name-emans.
  + A list of DC Names that map to IP addresses that map to site-name-emans and domain names.
  + A list of pre-proc-blocks that map to site-name-pre-procs.
  + A list of pre-proc-blocks that are to be ignored.
  + A list of site-link-pre-procs and associated site-link-costs.
  + A list of site-name-temps associated with:
    - A router-name.
    - A list of subnet-lans.
    - A list of site-link-halfs.
      + Each site-link-half is associated with a subnet-link.
    - A list of subnet-ip-routes.

* Stage 4
  Compare all subnet-lans, subnet-links and subnet-ip-routes with the pre-proc-blocks.
  - If a subnet is a subset of, superset of, or overlaps a pre-proc-block, print exception and discard the subnet. Print a specific exception message for each of the three conditions and specify the subnet and the pre-proc-block.
  - If a site-link-half is associated with a subnet-link that has been discarded, discard the site-link-half as well.
  - If a site-name-temp no longer contains any valid subnet-lans, discard it and any associated router-names, site-link-halfs, subnet-links and subnet-ip-routes.
  - At this point, we have:
    + The same results as at the end of Stage 3, with the exception of information that was removed because it was superceded by pre-processing information.
    + A list of site-name-emans.
    + A list of DC Names that map to IP addresses that map to site-name-emans and domain names.
    + A list of pre-proc-blocks that map to site-name-pre-procs.
    + A list of pre-proc-blocks that are to be ignored.
      - We are finished using the "ignore" pre-proc-blocks so we'll remove them from this list.
    + A list of site-link-pre-procs and associated site-link-costs.
    + A list of site-name-temps associated with:
      - A router-name.
      - A list of subnet-lans.
      - A list of site-link-halfs.
        + Each site-link-half is associated with a subnet-link.
      - A list of subnet-ip-routes.

* Stage 5
  Compare all subnet-ip-routes with all subnet-lans and subnet-links.
  - If a subnet-ip-route is a subset of, superset of, or overlaps a subnet-lan or subnet-link (should never happen), print an exception and discard the subnet-ip-route.
  - At this point, we have:
    + The same results as at the end of Stage 4, with the exception that some of the subnet-ip-routes were removed due to overlaps with other subnets.
    + A list of site-name-emans.

+ A list of DC Names that map to IP addresses that
  map to site-name-emans and domain names.
+ A list of pre-proc-blocks that map to site-name-pre-procs.
+ A list of site-link-pre-procs and associated site-link-costs.
+ A list of site-name-temps associated with:
  - A router-name.
  - A list of subnet-lans.
  - A list of site-link-halfs.
    + Each site-link-half is associated with a subnet-link.
  - A list of subnet-ip-routes.

* Stage 6
Compare IP addresses of DCs with subnet-ip-routes and
subnet-links.
- If a DC IP address fits inside a subnet-ip-route print an
  exception. This should probably be alarmed, since this
  could result in significant problems. What if I had a DC
  at home and so did someone else. If our subnets were merged
  into a single site, the two boxes would think they were
  connected via a LAN and try to replicate appropriately.
  For now, we'll let it happen just to see if this occurs.
- If a DC IP address fits inside a subnet-link, print an
  exception and abort.
- At this point, we have:
  + The same results as at the end of Stage 5. We may have
    generated some exceptions or we may have aborted.
  + A list of site-name-emans.
  + A list of DC Names that map to IP addresses that
    map to site-name-emans and domain names.
  + A list of pre-proc-blocks that map to site-name-pre-procs.
  + A list of pre-proc-blocks that are to be ignored.
  + A list of site-link-pre-procs and associated site-link-costs.
  + A list of site-name-temps associated with:
    - A router-name.
    - A list of subnet-lans.
    - A list of site-link-halfs.
      + Each site-link-half is associated with a subnet-link.
    - A list of subnet-ip-routes.

* Stage 7
Summarize remaining subnet-ip-routes, create a site-name-ip-route
and a site-link for each site-name-temp (per router at this point).
- Summarize all of the subnet-ip-routes associated with a
  site-name-temp, into the largest site-blocks possible.
- Create a new site-name-ip-route which includes all
  of the summarized subnet-ip-routes.
- Create a site-link between the newly created
  site-name-ip-route and the associated site-name-temp.
  Assign a site-link-cost of T1 (2717) to this site-link.
- Get rid of the associations between site-name-temps
  and subnet-ip-routes.
- At this point, we have:
  + A list of site-name-emans.
  + A list of DC Names that map to IP addresses that
    map to site-name-emans and domain names.
  + A list of pre-proc-blocks that map to site-name-pre-procs.
  + A list of site-link-pre-procs and associated site-link-costs.
  + A list of site-name-temps associated with:
    - A router-name.
    - A list of subnet-lans.
    - A list of site-link-halfs.
      + Each site-link-half is associated with a subnet-link.
    - A list of site-links to site-name-ip-routes.
  + A list of site-name-ip-routes with the following associations.
    - A list of subnet-ip-routes.
    - A site-link to a site-name-temp with a site-link-cost.

* Stage 8
  Compare subnet-lans with other all other subnet-lans. Look
  for incompatible masks, print exceptions and adjust. This
  will allow sites to still merge, even though they have
  incompatible masks.
  - Loop through all of the site-name-temps.
  - For each site-name-temp, loop through the subnet-lans.
  - If a subnet-lan is a subset of, superset of, or
    overlaps another subnet-lan, print an exception and
    change the mask of the smaller subnet to match that of
    the larger one.
  - At this point, we have:
    + The same results as at the end of Stage 7. We may have
      generated some exceptions and changed a few subnet masks.
    + A list of site-name-emans.
    + A list of DC Names that map to IP addresses that
      map to site-name-emans and domain names.
    + A list of pre-proc-blocks that map to site-name-pre-procs.
    + A list of site-link-pre-procs and associated site-link-costs.
    + A list of site-name-temps associated with:
      - A router-name.
      - A list of subnet-lans.
      - A list of site-link-halfs.
        + Each site-link-half is associated with a subnet-link.
      - A list of site-links to site-name-ip-routes.
    + A list of site-name-ip-routes with the following associations.
      - A list of subnet-ip-routes.
      - A site-link to a site-name-temp with a site-link-cost.

* Stage 9
  Combine multiple sites into complete sites for all
  site-name-temps. Move site-link-halfs and site-link-ip-routes
  to the newly created sites and generate a new site-name-temp.
  - Loop through all of the site-name-temps.
  - For each site-name-temp, loop through the subnet-lans.
  - If a subnet-lan matches one on any other site-name-temp,
    merge the two sites into a new site-name-temp.
  - Associate all site-link-halfs associated with the old
    site-name-temps with the new site-name-temp.
  - Move site-links to site-name-ip-routes from the old site-name-temp
    to the new site-name-temp.
  - Move router-names from old site-name-temps to new site-name-temp.
  - Delete the old site-name-temps.
  - Repeat this process until you can no longer combine sites.
  - At this point, we have:
    + Essentially the same as at the end of Stage 8, with the
      exception that there are fewer, but bigger, site-name-temps.
    + A list of site-name-emans.
    + A list of DC Names that map to IP addresses that
      map to site-name-emans and domain names.
    + A list of pre-proc-blocks that map to site-name-pre-procs.
    + A list of site-link-pre-procs and associated site-link-costs.
    + A list of site-name-temps associated with:
      - A list of router-names.
      - A list of subnet-lans.
      - A list of site-link-halfs.
        + Each site-link-half is associated with a subnet-link.
      - A list of site-links to site-name-ip-routes.
    + A list of site-name-ip-routes with the following associations.
      - A list of subnet-ip-routes.
      - A site-link to a site-name-temp with a site-link-cost.

* Stage 10
  Generate site-links between site-name-temps.
  - Loop through all of the site-name-temps.
  - For each site-name-temp, loop through the subnet-links.
  - If a subnet-link matches one on any other site-name-temp:

    + Check to see if we already have a site-link between
      the two site-name-temps. If so, create the site-link,
      but give it a unique name (i.e. site72-site38-2).
    + Ensure we don't create the same site-link twice. That
      is for the same WAN link between two sites, don't create
      one in each direction (i.e. site1-site2 and site2-site1).
    + Create a site-link between the two site-name-temps.
    + Check the bandwidth associated with each site-link-half.
    + If bandwidth matches, generate a site-link-cost based
      on the bandwidth and associate it with the new site-link.
    + If the bandwidth doesn't match, generate an exception.
      Generate a site-link-cost based on the lower of the
      two bandwidths and associate it with the new site-link.
    + If one site-link-half has a bandwidth, but the other
      doesn't, use the sole bandwidth for the site-link-cost.
    + If there is no bandwidth for either site-link-half,
      generate an exception and create a site-link-cost
      of X.
  - If a subnet-link matches more than one subnet-link on
    any other site-name-temp, generate an exception. Use
    the first match.
  - When finished connecting site-name-temps, loop through
    all remaining site-link-halfs and print an exception
    indicating we have a site-link-half that goes nowhere.
    Discard these site-link-halfs.
  - At this point, we have:
    + The same results as from Stage 9, except now most (if
      not all) site-name-temps are connected to other site-name-temps
      via site-links with appropriate site-link-costs. We have
      discarded all site-link-halfs from further processing.
    + A list of site-name-emans.
    + A list of DC Names that map to IP addresses that
      map to site-name-emans and domain names.
    + A list of pre-proc-blocks that map to site-name-pre-procs.
    + A list of site-link-pre-procs and associated site-link-costs.
    + A list of site-name-temps and site-name-ip-routes
      associated with:
      - A list of router-names. (Not for site-name-ip-routes)
      - A list of site-blocks.
      - A list of site-links to other site-names, site-name-temps
        and site-name-ip-routes with site-link-costs.

* Stage 11
  Summarize the subnet-lans in each of the site-name-temps to
  the largest site-blocks possible. When complete, compute the
  "largest site" by computing the total number of addresses
  in each site-name-temp.
  - At this point, we have:
    + Essentially the same as at the end of Stage 10, with the
      exception that the subnet-lans have been summarized into
      the largest possible blocks and we have computed the
      "largest site" based on total number of addresses.
    + A list of site-name-emans.
    + A list of DC Names that map to IP addresses that
      map to site-name-emans and domain names.
    + A list of pre-proc-blocks that map to site-name-pre-procs.
    + A list of site-link-pre-procs and associated site-link-costs.
    + A list of site-name-temps associated with:
      - A list of router-names.
      - A list of site-blocks.
      - A list of site-link-halfs.
        + Each site-link-half is associated with a subnet-link.
      - A list of site-links to site-name-ip-routes.
    + A list of site-name-ip-routes with the following associations.
      - A list of subnet-ip-routes.
      - A site-link to a site-name-temp with a site-link-cost.

* Stage 12
Replace site-name-temps with site-names where possible.
- Loop through the DCs and their IP addresses.
- For each DC IP address.
  + Loop through the site-name-temps.
  + For each site-block.
    - If the IP address of the DC falls inside the site-block, check if a site-name-eman has already been temporarily assigned for this site-name-temp.
    - If so, compare the two. If they are different, generate an exception and abort.
      If they are the same, just go to the next site-block.
  + When finished looping through the site-blocks, either there will have been a match for a site-name-eman or not.
  + If there is a match, rename the site from it's site-name-temp to the appropriate site-name-eman. Associate the DC and the domain with the site-name-eman.
  + If there was no match, retain the site-name-temp. This is a site with no DC.
- At this point, we have:
  + The same results as from Stage 11, except now some of the site-name-temps have been renamed with site-name-emans.
  + A list of site-name-emans.
  + A list of DC Names that map to IP addresses that map to site-name-emans and domain names.
  + A list of pre-proc-blocks that map to site-name-pre-procs.
  + A list of site-link-pre-procs and associated site-link-costs.
  + A list of site-name-emans, site-name-temps and site-name-ip-routes associated with:
    - A list of router-names. (Not for site-name-ip-routes)
    - A list of site-blocks.
    - A list of site-links to other site-name-emans, site-name-temps and site-name-ip-routes with site-link-costs.
  + For each site-name-eman, there is also an association with:
    - One or more DCs.
    - One or more domains.

* Stage 13
Merge the site-name-pre-procs with the site-name-emans.
- Loop through the site-name-pre-procs.
- For each site-name-pre-proc.
  + Loop through the site-name-emans.
  + If the site-name-eman and site-name-pre-proc match, merge the pre-proc-blocks associated with the site-name-pre-proc with the site-name-eman and remove the site-name-pre-proc.
- At this point, we have:
  + The same results as from Stage 12, except some of the pre-proc-blocks have been merged. It is still possible that some are left. These would be pre-proc-blocks that are associated with a site-name-pre-proc that did not map to a site-name-eman associated with a DC. These will either be linked to a site-name-eman via a site-link-pre-proc or will be an island.
  + A list of site-name-emans.
  + A list of DC Names that map to IP addresses that map to site-name-emans and domain names.
  + A list of site-link-pre-procs and associated site-link-costs.
  + A list of site-name-emans, site-name-temps and site-name-ip-routes associated with:
    - A list of router-names. (Not for site-name-ip-routes)
    - A list of site-blocks.
    - A list of site-links to other site-name-emans, site-name-temps and site-name-ip-routes with site-link-costs.
  + For each site-name-eman, there is also an association with:
    - One or more DCs.
    - One or more domains.

* Stage 14
  Merge the site-link-pre-procs with the site-links.
  - Loop through the site-link-pre-procs.
  - For each site-link-pre-proc.
    + Parse out the two site-name-pre-procs.
    + Loop through the site-name-emans.
    + If both site-name-pre-procs match a site-name-eman, then
      the site-link-pre-proc is a legitimate site-link.
      Check to see if a site-link already exists for these
      two sites. If so, print an exception and discard the
      site-link-pre-proc. (May not want to do this. Could
      either override any found or perhaps add it.) If not,
      create the site-link with the associated site-link-cost and
      discard the site-link-pre-proc.
    + If only one matches, check the one that doesn't match
      to see if it is associated with a site-name-pre-proc.
      If not, print an exception and discard the
      site-link-pre-proc.
    + If neither match, check both to see if they are associated
      with a site-name-pre-proc. If either one is not, print
      an exception and discard the site-link-pre-proc.
  - At this point, we have:
    + The same results as from Stage 13, except we have now
      finished all processing of the pre-processing data, so
      from now on, we can ignore it.
    + A list of site-name-emans.
    + A list of DC Names that map to IP addresses that
      map to site-name-emans and domain names.
    + A list of site-name-emans, site-name-temps, site-name-ip-routes
      and site-name-pre-procs associated with:
      - A list of router-names. (Not for site-name-ip-routes or
        site-name-pre-procs.)
      - A list of site-blocks (or pre-proc-blocks).
      - A list of site-links to other site-name-emans, site-name-temps,
        site-name-ip-routes and site-name-pre-procs with
        site-link-costs.
    + For each site-name-eman, there is also an association with:
      - One or more DCs.
      - One or more domains.

Stage 15
  We've now actually completed a full topology. All of the
  site-name-XXXs now all have the same set of attributes and
  so there is no real need to treat them seperately. We'll
  assume the name at this point will suffice to indicate the
  type of site it is. Examples:

    site-name-eman: SJ, RTP, AMS
    site-name-temp: site31, site6, site132
    site-name-ip-route: siteipr89, siteipr3, siteipr23
    site-name-pre-proc: sitepp27, sitepp9, sitepp167

  The topology created at this point is suitable for all
  NCs, assuming we have site transitivity turned on.

Stage XX
  Add replication schedules to site-links based on site-link-cost.

Perhaps we should take all subnet-lans at the islands and
merge them into the largest site. Another option would be to merge all islands into one
big site and then create a site-link back to SJC. This would be at least cause any island
site with a DC to use intersite replication instead of intrasite.

*****

* Stage 14
Generate the Island Reports
In order to be able to complete full replication for each NC, we need to be sure that we have full connectivity between all of the necessary sites. For the Schema and Configuration NCs, this is all sites. For the domain NC, this is all sites associated with each domain. The pre processing done in stage 1 is intended to ensure we have full connectivity at the end of Stage 13. However, it is possible due to a variety of reasons this is not the case and we will end up with "islands" of sites. So, we need to generate an island report for the full topology as well as for each domain. If we end up with more than one island for any of these, we'll need to create one or more site links for the preprocessing done in stage 1 to reduce the islands to only one.

The Island Report can be used to augment the preprocessing data used in stage 1, or it can be used to determine that we are missing configuration files, or have mis-configured information on the routers.

- Merge all site-name-emans, site-name-temps, site-name-ip-routes and site-name-pre-procs into a single list of site-names.
- Loop through the list of site-names.
- If the site-name is not associated with an island, create a new island and put the site-name on the island. If the site-name is a site-name-eman, then create a seperate island for each domain the site-name-eman is associated with. For each of the steps below, repeat for each domain. This will generate an Island Report for the full topology and for each domain.
- If the site-name has site-links, loop through the site-links.
  + Check to see if the remote site is associated with an island.
  + If not, then put it on the same island as the current site.
  + If its already on an island, check to see if it's island is different than that of the current island. If so, then take all site-names that are on the second island and put them on the current site-names island.
- Keep looping through all of the site-names until all are on islands.
- Some psuedo code for the island algorithm. Need to adapt for per domain topology.

```
if ($island{$site-name} == 0)     # site is not on an island
{
  $island++;                      # create a new island
  $island{$site-name} = $island; # Put the site on the island.
  $current-island = $island;      # set current island to be the new island
}
else                              # site is already on an island
{
  $current-island = $island{$site-name};  # set the current island to the
                                          # one associated with this site.
}
if ($site-name has site-links)
{
  foreach (site-link)         # loop through the site-links
  {
    $remote-site-name = site-name at the other end of the link;
    if ($island{$remote-site-name} == 0)  # remote-site-name not on an island
    {
      $island{$remote-site-name} = $current-island;
    }
    else                      # site-name is already on an island
    {
      if ($island{$remote-site-name} != $current-island)
      {
```

```
            # If we get here, we've basically found a link between
            # what used to be two islands.  We now need to merge
            # these into a single island.

            $old-island = $island{$remote-site-name}

            # Now loop through the sites.  Check if the island set
            # for each site has a different island

            foreach site
            {
               if $island{$site-name} == $old-island)
                  {$island{$site-name} = $current-island;}
            }
         }
      }
   }
}
```

- Print out the island report for the full topology and for each domain.
- For the full topology check if we have more than one island. If so, print out the island topology and abort. If we have only one island, we are in good shape. We can deal with multiple islands per domain in Stage 15.
- At this point, we have:
  + Onle one island for the full topology.
  + One or more islands for each domain.
  + A list of site-name-emans.
  + A list of DC Names that map to IP addresses that map to site-name-emans and domain names.
  + A list of site-name-emans, site-name-temps, site-name-ip-routes and site-name-pre-procs associated with:
    - A list of router-names. (Not for site-name-ip-routes or site-name-pre-procs.)
    - A list of site-blocks (or pre-proc-blocks).
    - A list of site-links to other site-name-emans, site-name-temps, site-name-ip-routes and site-name-pre-procs with site-link-costs.
  + For each site-name-eman, there is also an association with:
    - One or more DCs.
    - One or more domains.

This next stage is probably unnecessary since we'll have site transitivity enabled.

* Stage 15

Ensure that we have only one island for each domain. If a domain topology has more than one island, we need to create a site-link between islands. To do this right, we need to find the least cost path between sites.

- Loop through the domains.
- For each domain.
  + If there is only one island, skip to the next domain. We have a suitable replication topology for that domain.
  + If the domain has more than one island, loop through the islands. Start with the island with the least number of sites in it.
  + For each site-name on the island, get the list of site-links and their associated site-name-remote and site-link-cost.
  + If the site-name-remote is on the same island, discard the site-link. We now have a list of site-links and site-name-remotes that are on one or more different islands.
  + For each site-name-remote, get the list of site-links and their associated site-link-cost and site-name-remote.
    - Loop through each site-name-remote.
    - If the site-name-remote is the original site-name,

skip it. We only want site-links to other sites.
+ Loop through the next set of site-name-remotes.
- For each site-name-remote, see if it is associated with the current domain. If so, we've reached another "same domain island". Get the site-link-cost associated with this site-link. Add it to the site-link-cost for the site-link between the current site-name-remote and the original site. This is the site-link-cost for a direct site-link between the original site-name and the site-name-remote on the new same domain island.
-

Exception Reports

The program will need to generate the following exception reports. Reports should be posted on the web, as well as potentially emailed to responsible groups.

* Missing bandwidth. List of interfaces missing required bandwidth configurations.
* Unmatched bandwidth. Point to point links should have identical bandwidth configurations on both ends of the link. This report will list those links where the bandwidth statements don't match.
* Sites with no links. This report will consist of sites that have no links connecting them to other sites.
* Sites with only one connection. Lists those sites which have one end of a connection, but the other end does not terminate in another site.
* New interface type. Router interfaces are used to distinguish site interface from link interfaces. This report will list any interfaces which are not already appropriately categorized, usually due to new technology implementations.
* Old router configuration files. The EMAN DB needs to be checked for router configuration files with a "last successful download" date that is not current. Any file that is older than two days should be flagged as a potential problem.
* New sites. The algorithm should be able to detect new sites that contain routers that have no site name in EMAN. (This may not be a problem depending on how this is handled.)
* Probably need to check for configuration file information that may be old. Will involve checking the "last successful download" value in the EMAN database.
* IP Routes. For all the networks found via an "ip route" statement, we need to compare them with networks found via interfaces. If an ip route network matches, is a subset or a superset of an interface network, print an exception.
* interface FastEthernet1/0
ip address negotiated (error)

Actual Update process.

* Add sites
* Delete subnets
* Add subnets
* Modify subnets
* Add site links
* Modify site Links
* Delete site links
* Add servers
* Modify servers
* Delete sites

* No delete of servers?

Does Modify servers include renaming them? What happens when someone renames a server in EMAN? You just add it.
* Flag servers found in AD, but not in EMAN.